Tecnológico de Costa Rica

High Performance Embedded Systems

Project 3

Optimizing Traditional and Deep Learning Algorithms for Autonomous Driving

Professors

MSc. José Araya Martínez
MSc. Sergio Arriola-Valverde

Students

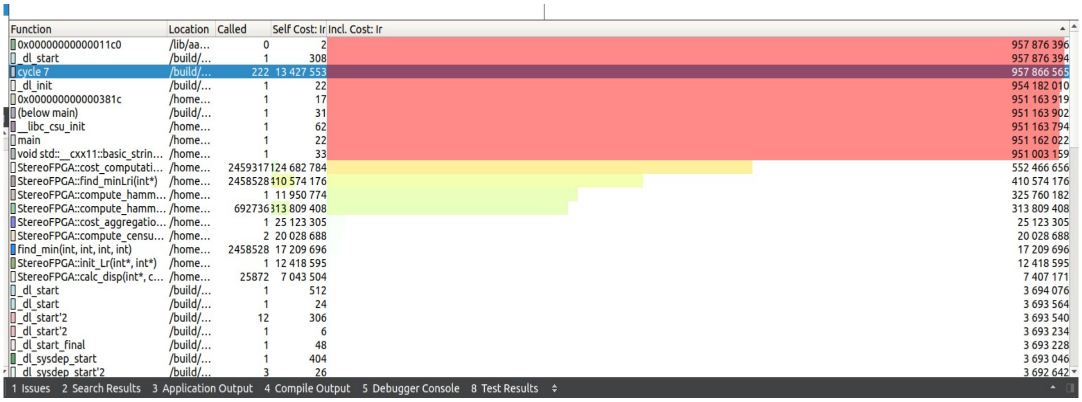
Sergio Guillén Fernández
Eliécer Mora Alaniz

First Third of year
2020

SGBM Algorithm Profiling

Given that the result obtained when attempting to profile with valgrind the provided implementation of the SGBM algorithm is that it takes a lot of time just to start the the Graphical User Interface (GUI) in the provided implementation, it was required to port the implementation to a console-based application. After implementing this, it was possible to obtain good results in the profiling.

Part of the most important results are presented in the Illustration 1.



Function	Location	Called	Self Cost: Ir	Incl. Cost: Ir
0x00000000000011c0	/lib/aa...	0	2	957 876 396
_dl_start	/build/...	1	308	957 876 394
cycle 7	/build/...	222	13 427 553	957 866 565
_dl_init	/build/...	1	22	954 182 010
0x000000000000381c	/home...	1	17	951 163 919
(below main)	/build/...	1	31	951 163 902
__libc_csu_init	/home...	1	62	951 163 794
main	/home...	1	22	951 162 022
void std::__cxx11::basic_strin...	/home...	1	33	951 003 159
StereoFPGA::cost_computati...	/home...	2459317	124 682 784	552 466 656
StereoFPGA::find_minLri(int*)	/home...	2458528	410 574 176	410 574 176
StereoFPGA::compute_hamm...	/home...	1	11 950 774	325 760 182
StereoFPGA::compute_hamm...	/home...	692736	313 809 408	313 809 408
StereoFPGA::cost_aggregatio...	/home...	1	25 123 305	25 123 305
StereoFPGA::compute_censu...	/home...	2	20 028 688	20 028 688
find_min(int, int, int, int)	/home...	2458528	17 209 696	17 209 696
StereoFPGA::init_Lr(int*, int*)	/home...	1	12 418 595	12 418 595
StereoFPGA::calc_disp(int*, c...	/home...	25872	7 043 504	7 407 171
_dl_start	/build/...	1	512	3 694 076
_dl_start	/build/...	1	24	3 693 564
_dl_start'2	/build/...	12	306	3 693 540
_dl_start'2	/build/...	1	6	3 693 234
_dl_start_final	/build/...	1	48	3 693 228
_dl_sysdep_start	/build/...	1	404	3 693 046
_dl_sysdep_start'2	/build/...	3	26	3 692 642

Illustration 1: Results after profiling the SGBM algorithm.

Given this, it was considered that the three most computation-expensive functions are:

- cost_computation.
- find_minLri.
- compute_hamming (really close with compute_hamming_distance).

Memory optimization

Once the algorithm was profiled, the next step was to optimize the amount of memory based on some assumptions in the input parameters.

Assuming that the input parameters are restricted to:

- BlockSize: 5 pixels
- P2 : 128
- NumDir: 4
- $C_{\text{máx}}$ (p, d): Results from Equation 3

The following results are obtained:

For equation 2:

$$\text{Census Transform Bit depth} = \text{Blocksize} * \text{Blocksize} - 1$$
$$= 5 * 5 - 1$$
$$= 24$$

For equation 3:

$$C_{\text{máx}}(p, d) = \text{Log}_2\,(\text{Census Transform Bit depth})$$
$$= \text{Log}_2(24)$$
$$= 4.58$$

For equation 4:

$$L \leq C_{\text{máx}}(p, d) + P2$$
$$L \leq 4.58 + 128$$
$$L \leq 132.58$$

For equation 6:

$$S \leq \text{Num. Dir} * (C_{\text{máx}}(p, d) + P2)$$
$$S \leq 4 * 132.58$$
$$S \leq 530.32$$